

On a class of $O(n^2)$ problems in computational geometry [☆]

Anka Gajentaan, Mark H. Overmars ^{*}

Department of Computer Science, Utrecht University, P.O. Box 80.089, 3508 TB Utrecht, Netherlands

Communicated by Emo Welzl; submitted 5 May 1993; accepted 14 October 1994

Abstract

There are many problems in computational geometry for which the best known algorithms take time $\Theta(n^2)$ (or more) in the worst case while only very low lower bounds are known. In this paper we describe a large class of problems for which we prove that they are all at least as difficult as the following base problem 3SUM: Given a set S of n integers, are there three elements of S that sum up to 0. We call such problems 3SUM-hard. The best known algorithm for the base problem takes $\Theta(n^2)$ time. The class of 3SUM-hard problems includes problems like: Given a set of lines in the plane, are there three that meet in a point?; or: Given a set of triangles in the plane, does their union have a hole? Also certain visibility and motion planning problems are shown to be in the class. Although this does not prove a lower bound for these problems, there is no hope of obtaining $o(n^2)$ solutions for them unless we can improve the solution for the base problem.

Keywords: Incidence; Separator; Covering; Visibility; Motion planning; 3SUM-hard problems

1. Introduction

The process of designing efficient algorithms for geometric problems can be quite frustrating. Sometimes you spend large portions of time trying to improve time-bounds that should be improvable but without any success. In such cases it helps if you know that the problem you are trying to tackle belongs to some class of “difficult” problems. A well-known class of such problems is the class of NP-complete problems. In computational geometry there are though many other barriers in complexity that seem difficult to pass. One such a barrier is $\Theta(n^2)$. There are many problems for which the

[☆] This work was partially supported by the ESPRIT III Basic Research Action No. 7141 (project ALCOM II) and by the Netherlands Organization for Scientific Research (N.W.O.).

^{*} Corresponding author.

best known algorithms take time $\Theta(n^2)$ (or more) in the worst case while only very low lower bounds are known.

In this paper we describe a large class of so-called 3SUM-hard problems¹ for which we prove that they are all at least as hard as the following base problem: Given a set S of n integers, are there three elements of S that sum up to 0. The best known algorithm for this base problem takes $\Theta(n^2)$ time. We define a notion of reduction and prove that the base problem can be reduced to a large number of seemingly unrelated geometric problems. The list of problems includes among others:

- Given a set of lines in the plane, are there three that pass through the same point?
- Given a set of (non-intersecting, axis-parallel) line segments, is there a line that separates them into two non-empty subsets?
- Given a set of (infinite) strips in the plane, do they fully cover a given rectangle?
- Given a set of triangles in the plane, compute their measure.
- Given a set of horizontal triangles in space, can a particular triangle be seen from a particular viewpoint?
- Given a set of (non-intersecting, axis-parallel) line segment obstacles in the plane, and a rod, can the rod be moved, allowing translation and rotation, from a given source to a given destination without colliding with the obstacles?
- Given a set of (horizontal) triangle obstacles in space, and a vertical rod, can the rod be moved, allowing translation only, from a given source to a given destination without colliding with the obstacles?

Also many related versions of the problems mentioned fall in the class. Because the base problem can be reduced to all of these problems, none of them can be solved in time $o(n^2)$ unless a subquadratic solution exists for the base problem. Note that this does not mean that all problems are equivalent (like for NP-complete problems). It only means that they are at least as hard as the base problem. Also being 3SUM-hard does not mean that a subquadratic solution is impossible. It only says that one better first try to solve the base problem faster. Moreover, almost any lower bound for the base problem will immediately carry over to the 3SUM-hard problems. Recently, Erickson and Seidel [11] have obtained a $\Omega(n^2)$ lower bound for the base problem but under a weak model of computation. This result clearly gives an important indication but, because of its weakness, some of the 3SUM-hard problems discussed in this paper cannot even be solved in their model. Throughout this paper we assume as a model of computation a real RAM, although most reductions also apply for weaker models.

This paper is organized as follows. In the next section we formally define our notion of reduction and indicate what relation for lower- and upper bounds it establishes between problems. In Section 3 we define three equivalent versions of the basis 3SUM problem that are used for reductions. Next, in Sections 4 till 8, we study different types

¹ In a previous version of this paper the term n^2 -hard was used instead of 3SUM-hard. A number of papers [1,2,4,10,11,18] refer to them as such. It was though pointed out that n^2 -hard is a confusing term in the light of the notions used in complexity theory. It is highly unlikely that the problems are really hard in the usual sense. For example, in [1] it is pointed out that this would imply that $P \neq NP$.

of 3SUM-hard problems: incidence problems, separator problems, covering problems, visibility problems, and motion planning problems. Finally, in Section 9 we summarize our results, give some further conclusions and indicate directions for future research.

2. Reductions

Reductions will play a crucial role in the sequel of this paper. Our goal is to show relations between the complexity of particular problems. Rather than using the standard terminology used in for example the theory of NP-complete problems we will introduce different notations and terminology because we think they are better suited for our problems and gives more intuition. In particular, we will use the following notion of solvability.

Definition 2.1. Given two problems PR1 and PR2 we say that PR1 is $f(n)$ -solvable using PR2 iff every instance of PR1 of size n can be solved using a constant number of instances of PR2 of at most linear size and $O(f(n))$ additional time. We denote this as

$$PR1 \lll_{f(n)} PR2.$$

If $PR1 \lll_{f(n)} PR2$ and $PR2 \lll_{f(n)} PR1$ we say that PR1 and PR2 are $f(n)$ -equivalent, denoted as

$$PR1 = =_{f(n)} PR2.$$

When PR1 is $f(n)$ -solvable using PR2 this in some sense means that PR1 is “easier” than PR2. The following lemma makes this more explicit.

Lemma 2.1. Let $PR1 \lll_{f(n)} PR2$. Let $f(n)$ and $g(n)$ be polynomials. If PR2 can be solved in $O(g(n))$ time and $f(n) = O(g(n))$ then PR1 can be solved in $O(g(n))$ time. Reversely, if $\Omega(g(n))$ is a lower bound for PR1 and $f(n) = o(g(n))$ then $\Omega(g(n))$ is also a lower bound for PR2.

Proof. Follows immediately from the definition. \square

Hence, when $f(n)$ is sufficiently small, lower bounds for PR1 carry over to PR2 and upper bounds for PR2 hold for PR1.

3. The base problem

We define the following base problem.

Problem: 3SUM

Given a set S of n integers, are there $a, b, c \in S$ with $a + b + c = 0$?

The best algorithm known for this problem takes $\Theta(n^2)$ time in the worst case (see below). In the sequel of this paper we will prove that many problems are at least as hard as this base problem. Such problems we will call 3SUM-hard.

Definition 3.1. We call a problem PR 3SUM-hard if and only if 3SUM is $f(n)$ -solvable using PR, where $f(n) = o(n^2)$.

From Lemma 2.1 it immediately follows that when a problem PR is 3SUM-hard it is impossible to find a subquadratic algorithm for it unless a subquadratic algorithm for 3SUM exists. Also, an $\Omega(n^2)$ lower bound for 3SUM would immediately imply an $\Omega(n^2)$ lower bound for PR. Actually, because in all our reductions $f(n)$ is only $O(n \log n)$, even much smaller lower bounds will immediately carry over.

To be able to prove reductions we give the following equivalent base problem.

Problem: 3SUM'

Given three sets of integers A , B , and C of total size n , are there $a \in A$, $b \in B$ and $c \in C$ with $a + b = c$?

Theorem 3.1. $3SUM' =_n 3SUM$.

Proof. $3SUM \lll_n 3SUM'$ is easy to prove. Simply set $A = S$, $B = S$ and $C = -S$. Now obviously when for $a \in A$, $b \in B$ and $c \in C$ $a + b = c$ then $a, b, (-c) \in S$ and $a + b + (-c) = 0$.

The reverse is slightly more complicated. We want to create one set S such that whenever three elements in S add up to 0 there were three elements $a \in A$, $b \in B$, and $c \in C$ such that $a + b = c$. Without loss of generality we assume that all elements in the sets are positive. (If not, take a large enough number k and add k to each element in A and B , and add $2k$ to each element in C .) Let $m = 2 \cdot \max(A, B, C)$. Construct S as follows: For each element $a \in A$ put $d = a + m$ in S . For each element $b \in B$ put $b' = b$ in S . Finally, for each element $c \in C$ put $c' = -c - m$ in S . Clearly, if $a + b = c$ then $d + b' + c' = 0$. What remains to be shown is that whenever there are three elements in S that add up to 0 they came from three different sets.

From the construction of S it is immediately clear that for all $a \in A$, $b \in B$ and $c \in C$ we have $m < d \leq 1.5m$, $0 < b' \leq 0.5m$ and $-1.5m \leq c' < -m$. Let $x, y, z \in S$ with $x + y + z = 0$. At most one of the three elements can originally come from A because otherwise the sum would be at least $2m - 1.5m = 0.5m$. Similarly only one element can come from C because otherwise the sum would be less than $-2m + 1.5m = -0.5m$. Also one element must come from C because all elements that come from A and B are positive. Now let one element come from C and the other two from B . In this case the sum is smaller than 0 because the two elements from B add up to at most m and the element from C is smaller than $-m$. This leaves only one case namely that all three elements came from different sets. \square

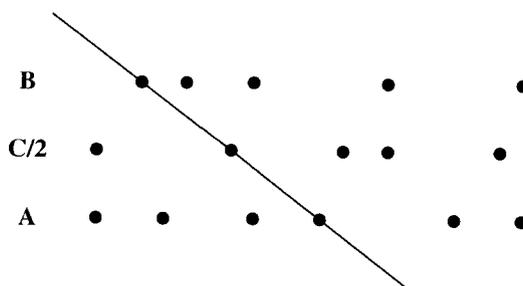


Fig. 1. An example of GEOMBASE.

Problem 3SUM' (and, hence, 3SUM) can be solved in $O(n^2)$ time as follows: Sort sets B and C . Now for each element $a \in A$ compute the sorted set $B + a$ (that is, the set of all number in B with a added to it) in $O(n)$ time. Next check, again in $O(n)$ time, whether $B + a$ and C have an element in common by a simultaneous traversal of the ordered sets. This takes another $O(n)$ time. Because we have to repeat this for all $O(n)$ elements $a \in A$ the total time required is bounded by $O(n^2)$.

Both base problems are non-geometric in nature. We will now define a geometric interpretation of 3SUM' because this will make reductions easier.

Problem: GEOMBASE

Given a set of n points with integer coordinates on three horizontal lines $y = 0$, $y = 1$, and $y = 2$, determine whether there exists a non-horizontal line containing three of the points.

Theorem 3.2. $\text{GEOMBASE} =_n \text{3SUM}'$.

Proof. We first show that $\text{3SUM}' \lll_n \text{GEOMBASE}$. For each element $a \in A$ we create a point $(a, 0)$ on the line $y = 0$. Similarly, for each element $b \in B$ we create a point $(b, 2)$ and for each element $c \in C$ we create a point $(c/2, 1)$. See Fig. 1. It is immediate that three points $(a, 0)$, $(b, 2)$ and $(c/2, 1)$ are collinear if and only if $a + b = c$. The reverse is proven in exactly the reverse way: for each point $(a, 0)$ create an element $a \in A$, for each point $(b, 2)$ create an element $b \in B$ and for each point $(c, 1)$ create an element $2c \in C$. \square

4. Incidence problems

A number of problems in computational geometry can be formulated as incidence problems. These are problems that, given a set of objects, ask whether there is an object from a particular class that is incident with (or intersects) at least some given number of objects in the set. For example, the well-known line segment intersection detection problem asks whether there is a point in the plane that is incident with at least two line segments in a set.

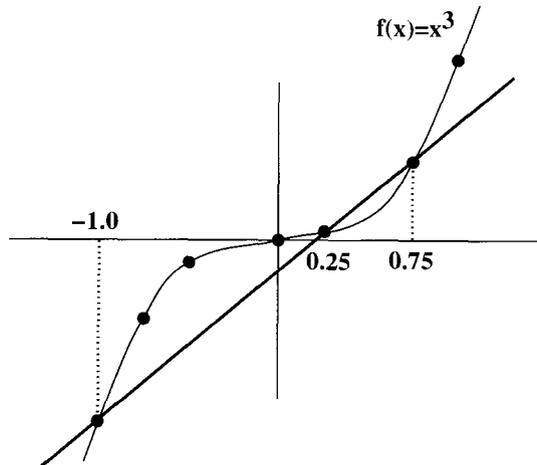


Fig. 2. Transforming 3SUM to 3-POINTS-ON-LINE.

When the number of incident objects is two the problem can normally be solved very efficiently. For example the above mentioned line segment intersection detection problem can be solved in time $O(n \log n)$ (see for example [19]). When we ask for incidence with at least three objects the problems tend to become a lot harder and many turn out to be 3SUM-hard.

Problem: 3-POINTS-ON-LINE

Given a set of points in the plane, is there a line that contains at least three of the points?

Theorem 4.1. $3\text{-SUM} \lll_n 3\text{-POINTS-ON-LINE}$.

Proof. We are given a set S of n integers. Each element $x \in S$ we transform into the point (x, x^3) . See Fig. 2 for an example. The claim is that $a + b + c = 0$ if and only if the three corresponding points (a, a^3) , (b, b^3) and (c, c^3) are collinear. This can be proven by some elementary calculations. \square

Of course 3-POINTS-ON-LINE is a special case of the more general problems that ask whether there exists a line that intersects at least three objects in a set of line segments, polygons, circles, etc. Hence, all these problems are 3SUM-hard as well.

Problem: POINT-ON-3-LINES

Given a set of lines in the plane, is there a point that lies on at least three of them?

This problem is the exact dual (using the right type of dualization) of 3-POINTS-ON-LINE. Hence, it immediately follows that

Theorem 4.2. $\text{POINT-ON-3-LINES} = =_n 3\text{-POINTS-ON-LINE}$.

Again many generalized versions of POINT-ON-3-LINES exist by asking for example whether there is a point incident to at least three objects in a set of line segments, polygons or circular arcs, because lines are a special case of all of them. Hence, all these problems are also 3SUM-hard. (Note that this might not be true for for example the problem whether there is a point incident to three objects in a set of disks.)

Most problems mentioned above can also easily be solved in $O(n^2)$. This is done by, either in the primal or dual, constructing the entire arrangement formed by the objects. This takes time $O(n^2)$ using topological sweeping (see [7]). Now the incidence must occur at a vertex of the arrangement which is easily checked in $O(n^2)$ time by traversing the arrangement.

An interesting extension to look at is the incidence problem for $k > 3$ objects. Although at first sight this might seem harder to solve it turns out that often the problem gets more easy. For example in [13] it is shown that the problem that asks for a line incident to at least k points can be solved in time $O((n^2/k) \log n/k)$ which is much faster than $O(n^2)$ if k grows with n .

5. Separator problems

Separator problems form, at first sight, a quite different class of problems. A separator is formally defined as follows.

Definition 5.1 Given a set S of n objects in the plane, we call a line l a separator of S if l does not intersect any object in S and both halfplanes bounded by l contain a non-empty subset of the objects in S .

Separators have been studied in several recent papers. See for example [12] and [10]. In this section we will show that many types of separator problems are 3SUM-hard. The first version is the following.

Problem: SEPARATOR1

Given a set S of n possible half-infinite, closed horizontal line segments, is there a non-horizontal separator?

Theorem 5.1. GEOMBASE $\lll_{n \log n}$ SEPARATOR1.

Proof. We are given a set of points with integer coordinates on three horizontal lines $y = 0$, $y = 1$ and $y = 2$. Let the x -coordinates of the points on the first line A , ordered from left to right be a_1, a_2, \dots, a_i . Similarly, let the points on the other lines B and C be b_1, \dots, b_j and c_1, \dots, c_k . Let $\varepsilon = 1/4$. Now transform the points on A into horizontal segments on A with x -intervals $[-\infty : a_1 - \varepsilon]$, $[a_1 + \varepsilon : a_2 - \varepsilon]$, \dots , $[a_i + \varepsilon : \infty]$. Similarly for the sets B and C . Clearly this transformation can be done in time $O(n \log n)$. It is obvious that, when there is a line through points a on A , b on B and c

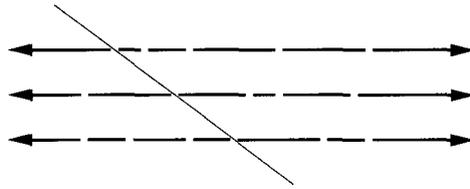


Fig. 3. Transforming GEOMBASE to SEPARATOR1.

on C , a non-horizontal separator exists. What remains is to prove the reverse. See Fig. 3 for the situation obtained for the example in Fig. 1. So let l be a non-horizontal separator. It is clear that l must run through three “holes” $(a - \epsilon : a + \epsilon)$ on A , $(b - \epsilon : b + \epsilon)$ on B and $(c - \epsilon : c + \epsilon)$ on C . Let the line intersect the holes at positions $(a + \delta_1)$, $(b + \delta_2)$ and $(c + \delta_3)$. So $(a + \delta_1) + (b + \delta_2) = 2(c + \delta_3)$ for δ_1 , δ_2 and δ_3 between $-\epsilon$ and ϵ . Because $\epsilon = 1/4$ and a , b and c are integers this is only possible when $a + b = 2c$, that is, when there is a line through points on A , B and C . \square

The same result holds for open line segments. The construction used in the proof will be used at a number of other places in the sequel.

You might not like the fact that the description of SEPARATOR1 allows for infinite segments and that the separator is not allowed to be horizontal. This can easily be remedied by adding some vertical line segments:

Problem: SEPARATOR2

Given a set S of n closed, non-intersecting (nor touching), axis-parallel line segments, is there a separator?

Theorem 5.2. $GEOMBASE \lll_{n \log n} SEPARATOR2$

Proof. Consider the construction shown in Fig. 4. We use the same set of segments as in the proof of Theorem 5.1, except that we replace the infinite segments by very long segments, with vertical segments at the end (note that they are non-symmetrical). It can easily be proven that, when we make the segments long enough and place the vertical segment near enough to them, no separator can run through the holes at the far left or far

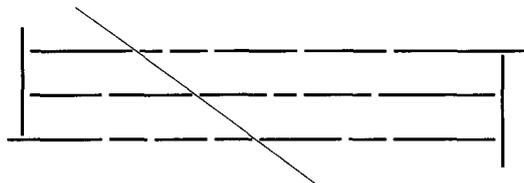


Fig. 4. Transforming GEOMBASE to SEPARATOR2.

right. Hence, using the same argument as above, a separator exists iff three points are collinear. \square

Clearly the problems above can be extended by asking for separators among other classes of objects like arbitrarily oriented line segments, possibly intersecting line segments, axis-parallel (or arbitrarily oriented) non-intersecting (or possibly intersecting) rectangles or triangles. All these problems have SEPARATOR2 as a special case and, hence, are 3SUM-hard. Also, all these problems can be solved in $O(n^2)$ time by considering the arrangement obtained when dualizing the objects.

6. Covering problems

Another type of geometric problems are covering problems. Here one asks whether the union of a set of objects fully covers (that is, contains) a particular object. We will look at some variations of this problem. Also we look at some related problems like the measure problem.

6.1. Covering a box or triangle

One of the simplest covering problems that is already 3SUM-hard is the strip cover problem. Let a strip be defined as the (infinite) area between two parallel lines.

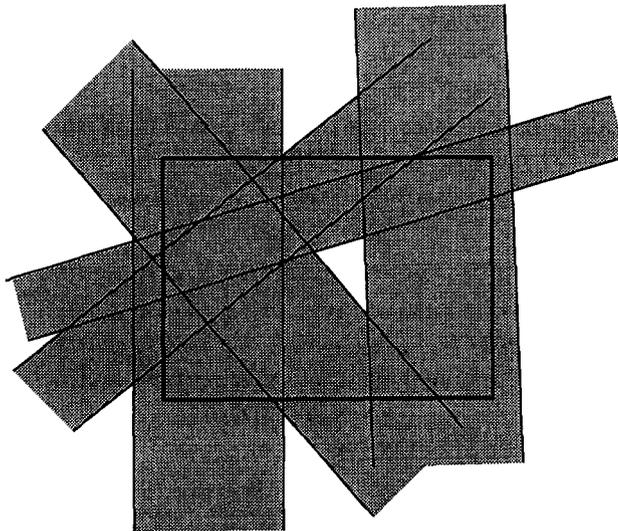


Fig. 5. The rectangle is not fully covered by the strips.

Problem: STRIPS-COVER-BOX

Given a set of strips in the plane, does their union contain a given axis-parallel rectangle?

See Fig. 5 for an example of this problem.

Theorem 6.1. $\text{GEOMBASE} \lll_n \log n \text{ STRIPS-COVER-BOX}$.

Proof. First we transform the instance of the GEOMBASE problem into a set of horizontal segments as in the proof of Theorem 5.1. We rotate this set to obtain a set of vertical line segments and dualize them using the point-line duality of [6]. Using this duality vertical line segments turn into strips. The 6 half-infinite segments turn into halfplanes. A non-vertical line in the primal plane transforms to a point in the dual. Such a line misses all the segments if and only if the point lies in none of the strips (or halfplanes).

Let P be the polygon obtained by taking the intersection of the complements of the six halfplanes. P corresponds to all lines that do not intersect any of the infinite segments. It can easily be verified that P is bounded. Now take an axis-parallel rectangle R that contains P . We turn each halfplane into a strip using a line outside R such that the intersection of the strip with R is the same as the intersection of the halfplane with R . Now consider the set of n strips thus obtained. If they fully cover R they fully cover P and, hence, there is no line separating the segments. On the other hand, if there is a point not covered inside R , it must lie inside P and it corresponds to a separating line. \square

It is obvious that this is a special case of the problem that asks whether a set of arbitrarily oriented rectangles covers a rectangle. Just cut off the strips outside the box, creating rectangles. Another version is the following:

Problem: TRIANGLES-COVER-TRIANGLE

Given a set of triangles in the plane, does their union contain another given triangle?

Theorem 6.2. $\text{STRIPS-COVER-BOX} \lll_n \text{ TRIANGLES-COVER-TRIANGLE}$.

Proof. We first show that $\text{STRIPS-COVER-BOX} =_n \text{ STRIPS-COVER-TRIANGLE}$ whose definition is clear. The second problem can solve the first by simply splitting the box into two triangles and checking whether both are covered. The reverse can be solved by taking a box around the triangle and adding three strips to the set that cover the area between the box and the triangle. Now the box is covered if and only if the triangle is covered.

It is also easy to see that $\text{STRIPS-COVER-TRIANGLE} \lll_n \text{ TRIANGLES-COVER-TRIANGLE}$. Simply cut each strip outside the triangle to obtain a rectangle and split each rectangle into two triangles. Now the triangles cover the triangle if and only if the strips cover the triangle. \square

Clearly one can again define many extensions of these simple covering problems that, hence, are also 3SUM-hard.

6.2. Union and measure

We will now look at two important related problems:

Problem: HOLE-IN-UNION

Given a set of triangles in the plane, does their union contain a hole?

Theorem 6.3. TRIANGLES-COVER-TRIANGLE \lll_n HOLE-IN-UNION.

Proof. Let S be the set of triangles and t the triangle to be covered. For each triangle $t' \in S$ cut off the piece lying outside t and, if the remaining part is not a triangle, split that part into triangles. In this way we obtain, in linear time, a set S' of $O(n)$ triangles that all lie fully inside t and whose union is the same as the union of S , restricted to t . Hence t is fully covered by S if and only if it is fully covered by S' . This is the case if and only if the union of S' does not contain a hole. \square

With a bit more effort we can also show the reverse and, hence, both problems are actually $(n \log^2 n)$ -equivalent:

Theorem 6.4. HOLE-IN-UNION $\lll_{n \log^2 n}$ TRIANGLES-COVER-TRIANGLE.

Proof. Let S be the set of triangles of which we want to determine whether their union has a hole. Determine a triangle t containing all the triangles in S . Next determine the outer contour C of S , being the part of the boundary of the union that bounds the outer face. See Fig. 6. This can be done in time $O(n \log^2 n)$ (see [8]). Finally triangulate the part between t and C which can be done in linear time (see [5]) and add these triangles to S , obtaining a set S' . Now it is clear that the union of S contains a hole if and only if S' does not cover t . \square

Another related problem is the following.

Problem: TRIANGLE-MEASURE

Given a set of triangles in the plane, compute the measure of their union.

Theorem 6.5. TRIANGLES-COVER-TRIANGLE \lll_n TRIANGLE-MEASURE.

Proof. Like in the proof of Theorem 6.3 we transform the set S of triangles into an equivalent set S' that lies fully inside t . Now it is immediate that t is fully covered if and only if the measure of the union of S' is the same as the measure of t . \square

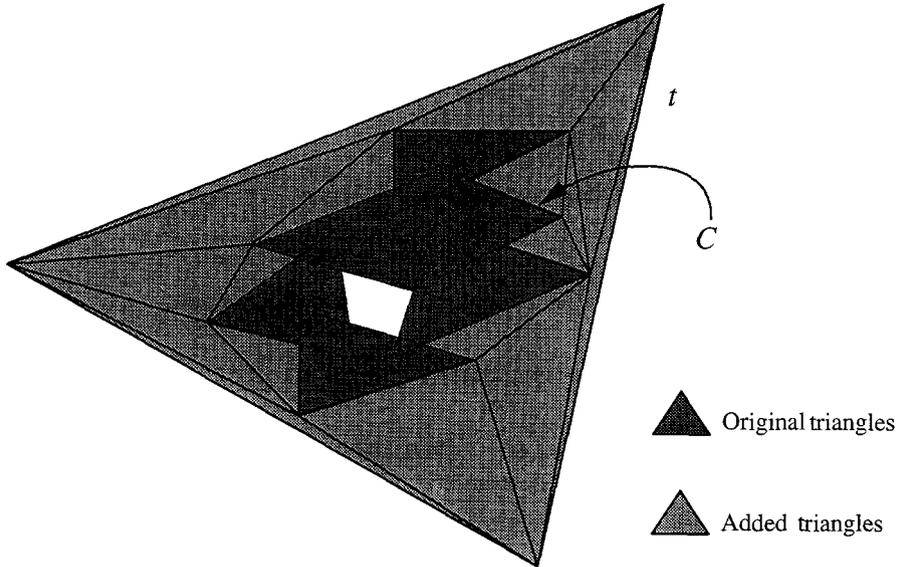


Fig. 6. Transforming HOLE-IN-UNION to TRIANGLES-COVER-TRIANGLE.

All the above problems can be solved in time $O(n^2)$ by considering the arrangement formed by the strips, rectangles or triangles. Again many extensions can be formulated that are 3SUM-hard as well.

6.3. Point covering

In this subsection we consider one other type of covering problem:

Problem: POINT-COVERING

Given a set of n halfplanes and a number k , determine whether there is a point p that is covered by at least k of the halfplanes.

Note that this problem is trivial for $k \leq n/2$. The answer is simply yes. But for larger k the problem becomes 3SUM-hard.

Theorem 6.6. STRIPS-COVER-BOX \lll_n POINT-COVERING.

Proof. For each strip take the two halfplanes that cover the complement of the strip. In this way we obtain a set S of $2n$ halfplanes. Clearly, any point that lies in none of the strips lies in exactly n of the halfplanes. Any other point lies in fewer halfplanes. Add to S the four halfplanes that point inwards from the four sides of the box. Now it is obvious that the strips do not cover the box if and only if there is a point covered by $n + 4$ halfplanes. \square

7. Visibility problems

Visibility problems are another important type of geometric problems. In general they ask which objects in a set can be seen from a particular point or which pairs of objects can see each other in a set. Here two points are said to see each other if the open line segment connecting them does not intersect any object. Two objects see each other if there exist points on them that see each other. In this section we will show that two rather simple versions of these problems are already 3SUM-hard. First we consider a problem in the plane. Next we look at three-dimensional problems.

7.1. Planar problems

Visibility from a point in the plane is normally easy. See for example [9]. Visibility from a line segment though is much harder. We formulate the following easy version of this problem:

Problem: VISIBILITY-BETWEEN-SEGMENTS

Given a set S of n horizontal line segments in the plane and two particular horizontal segments s_1 and s_2 , determine whether there are points on s_1 and s_2 that can see each other, that is, such that the open segment between the points does not intersect any segment in S .

Theorem 7.1. $\text{GEOMBASE} \lll_{n \log n} \text{VISIBILITY-BETWEEN-SEGMENTS}$.

Proof. Consider the set of segments used in the proof of Theorem 5.1. Place segments s_1 and s_2 just above and below the set of holes (see Fig. 7). It is clear that s_1 and s_2 can see each other if and only if there are three collinear holes, which proves the theorem. Note that it is no longer necessary to have the segments at the left and right extend all the way to infinity. \square

This result can of course be extended to other types of objects. Also the following version is 3SUM-hard.

Problem: VISIBILITY-FROM-INFINITY

Given a set S of axis-parallel line segments in the plane and one particular horizontal segments s , determine whether there is a point on s that can be seen from infinity, that

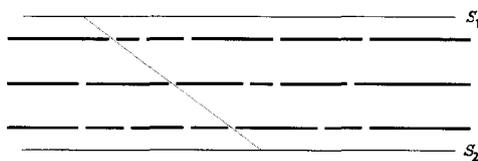


Fig. 7. Transformation from GEOMBASE to VISIBILITY-BETWEEN-SEGMENTS.

is, whether there exists an infinite ray starting at the point on s that does not intersect any segment.

Theorem 7.2. $\text{GEOMBASE} \lll_n \log n \text{ VISIBILITY-FROM-INFINITY}$.

The proof is left as an exercise to the reader.

All these two-dimensional visibility problems can be solved in time $O(n^2)$ by computing the visibility graph of the endpoints of the segments and extending the visibility edges in both directions until they hit a segment. Visibility normally runs along such an extended edge. An extended visibility graph can easily be computed in time $O(n^2)$ using the algorithm in [23].

7.2. Three dimensional visibility

Visibility problems in 3-dimensional space are normally harder than in the plane. As we will see visibility from a point can already be 3SUM-hard.

Problem: VISIBLE-TRIANGLE

Given a set S of opaque horizontal triangles, another horizontal triangle t and a viewpoint p , is there a point on t that can be seen from p ?

See Fig. 8 for an example of the problem. The problem can be solved in time $O(n^2)$ by computing all parts of the triangles that are visible, that is, by performing hidden surface removal. See for example [17].

Theorem 7.3. $\text{TRIANGLES-COVER-TRIANGLE} = =_n \text{ VISIBLE-TRIANGLE}$.

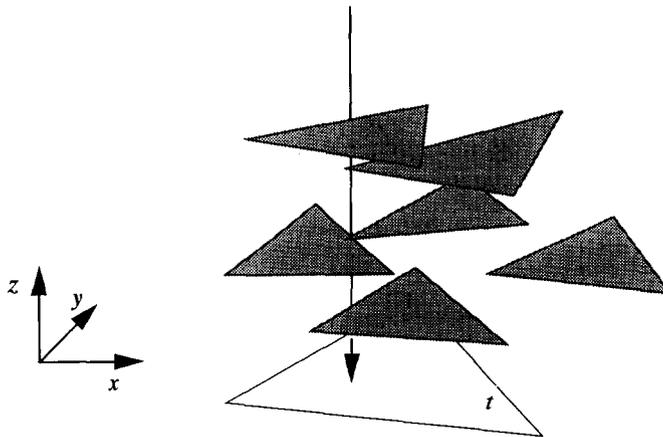


Fig. 8. Triangle t is visible.

Proof. First observe that, using standard perspective transformations VISIBLE-TRIANGLE can be mapped to the problem where we look down from infinity, which is, hence, equivalent.

We first show that TRIANGLES-COVER-TRIANGLE \lll_n VISIBLE-TRIANGLE. Let t be the triangle to be covered. Place t on the xy -plane. Put all other triangles on different heights above t . Then clearly t is partially visible from infinity if and only if the triangles do not fully cover t .

For the reverse, remove all triangles below t . Put the projections of the remaining triangles on the xy -plane in the set S . Also project t on the xy -plane. Now again t is partially visible from infinity if and only if the triangles in S do not fully cover t . \square

Also other 3-dimensional visibility problems can be shown to be 3SUM-hard. See for example [3].

8. Motion planning problems

Motion planning problems have been studied extensively in computational geometry because of the important applications to robotics. The motion planning problem asks for computing a path for a robot from a given source to a given goal configuration while avoiding collision with a set of obstacles. One should make a distinction here between the find-path problem that asks for a path and the existence problem that only asks whether a path exists. Many different solutions to the motion planning problem are known. See [14] for an overview. The complexity of such solutions tends to depend on the number of degrees of freedom. For example, a rigid robot that moves in the plane, allowing for both translation and rotation, has three degrees of freedom. In this section we will show that already some simple versions of the motion planning problem with three degrees of freedom are 3SUM-hard, even if we only ask for the existence of a path.

8.1. Planar motion planning

We first consider motion planning in the plane with three degrees of freedom. We formulate the following instance of this problem which is almost the easiest possible.

Problem: PLANAR-MOTION-PLANNING

Given a set of non-intersecting, non-touching, axis-parallel line segment obstacles in the plane and a line segment robot (a rod or ladder), determine whether the rod can be moved (allowing both translation and rotation) from a given source to a given goal configuration without colliding with the obstacles.

The best known solution for this problem takes time $O(n^2)$ (see [22]).

Theorem 8.1. GEOMBASE $\lll_{n \log n}$ PLANAR-MOTION-PLANNING.

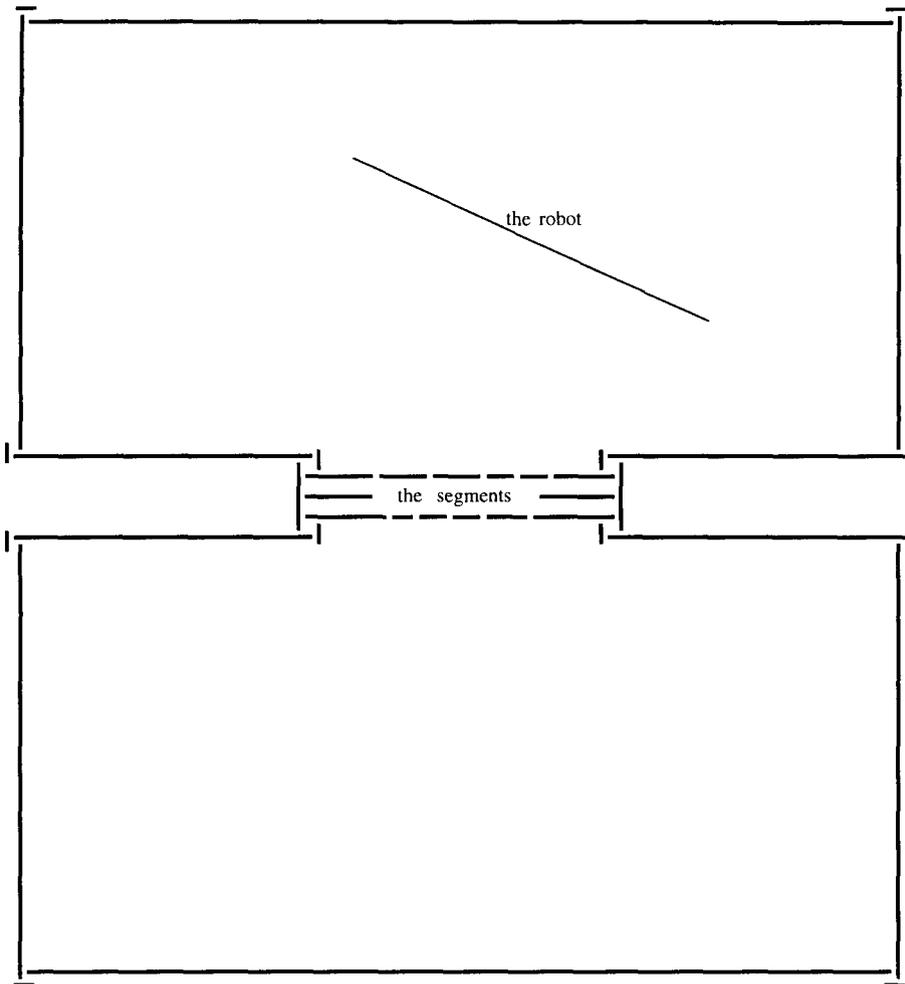


Fig. 9. Reduction from GEOMBASE to PLANAR-MOTION-PLANNING.

Proof. We will use the construction with horizontal line segments as described in the proof of Theorem 5.2. Only we change the left and right end and add some segments to obtain the structure shown in Fig. 9. We now want to move a long rod from the top region to the bottom region. It is easily verified that the rod can never escape from the two regions. Hence, it has to move among the segments in the center. Also, if the rod is long enough, it always has to be contained partially in either the top or in the bottom region. Hence, to move from top to bottom there must be a moment at which it is in both region. At this very moment the rod passes through three gaps among the segments and, by the same arguments as in the proof of Theorem 5.1, corresponds to a solution of GEOMBASE.

On the other hand, when a solution to GEOMBASE exists there is a motion for the rod from the top the the bottom region. Simply orient the rod in the direction of the three

holes and slide it from one side to the other. By making the two regions large enough this can be done without problem. So a solution to GEOMBASE exists if and only if a motion can be found. \square

Clearly, PLANAR-MOTION-PLANNING can be generalized in many ways. We can allow for more complicated types of obstacles like arbitrary line segments, triangles or axis-parallel rectangles, and we can allow for more general shapes of robots like rectangles or polygons. Also we can consider the find-path problem instead of the existence problem. All these version are also 3SUM-hard because they can be used to solve PLANAR-MOTION-PLANNING.

8.2. Translational motion planning in 3-space

We can also formulate a motion planning problem in 3-dimensional space that has three degrees of freedom by allowing for translational motion only. We will show that the following, rather simple, version is also 3SUM-hard:

Problem: 3D-MOTION-PLANNING

Given a set of horizontal (that is, parallel to the xy -plane) non-intersecting, non-touching triangle obstacles in 3-space, and a vertical line segment as a robot, determine whether the robot can be moved, using translations only, from a source to a goal position without colliding with the obstacles.

The best known solution to this problem takes time $O(n^2 \log n)$. This is based on the observation that the free space for this particular instance of the motion planning problem has complexity $O(n^2)$ (see [15]). Improvement to $O(n^2)$ might be possible.

Theorem 8.2. TRIANGLES-COVER-TRIANGLE \lll_n 3D-MOTION-PLANNING.

Proof. Let S be the set of triangles and let t be the triangle to cover. We will transform this to an instance of the motion planning problem where a unit length rod has to move from a position above the xy -plane to a position below that plane. We first want to enforce that the robot has to pass the xy -plane inside the triangle t . To this end we build a “cage” from which the robot cannot escape. Around the triangle t in the xy -plane we put three triangles that slightly overlap (see Fig. 10). To avoid the triangles to intersect we put them on slightly different heights. Such a structure we call a fence. It is clear that the robot can only escape from the fence by jumping over it or going under it. We repeat this fence at the height $z = -1$, $z = -0.5$, $z = 0.5$, and $z = 1$. Finally we place a large horizontal triangle (larger than t) at $z = -1.6$ and at $z = 1.6$. So we get five fences and a top and bottom triangle. In this way we create a triangular cage surrounding t that runs from $z = -1.6$ to $z = 1.6$ from which the robot cannot escape. See Fig. 10 for a picture of the cage.

Let the bottommost point of the robot be the reference point. We choose as source position for the robot a point in t but at $z = 0.5$ and as a goal position the same point at

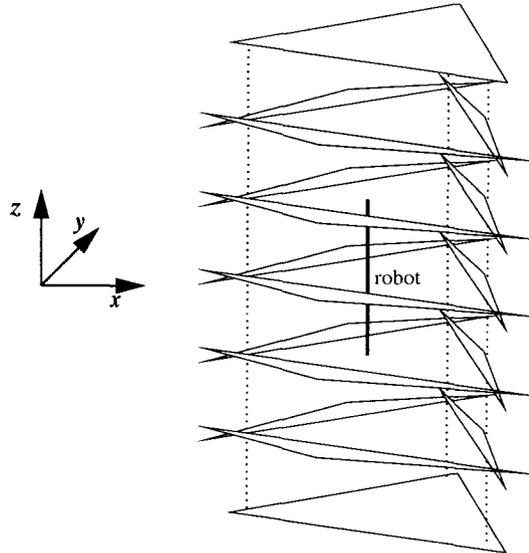


Fig. 10. A cage from which the robot cannot escape.

$z = -1.5$. So the robot starts fully above the xy -plane and ends fully below the xy -plane. Because it has to stay inside the cage it has to cross the xy -plane inside t .

Now consider the triangles in S . We place them all at slightly different heights with z -coordinate between 0 and 0.5. Clearly in this way we get $n + 17$ horizontal triangle obstacles that do not intersect (n from S and 17 for the cage). (Note that t itself does not occur in the set of obstacles.) It remains to show that a path for the robot exists if and only if the triangles in S do not fully cover t .

Let us first assume that the triangles in S do not fully cover t so there is a point (x, y) in t that is not covered. We can now move the robot from source to goal as follows: First move the robot horizontally from the source to $(x, y, 0.5)$. Next move it vertically down to $(x, y, -1.5)$. Finally move it horizontally to the goal. It is easily checked that this does not cause collisions.

Now assume a path from source to goal does exist. We have to show that in this case the triangles in S do not fully cover t . This can be seen as follows. At some moment during the motion the reference point of the robot lies in t on the xy -plane. Let this happen at some point (x, y) . Because the robot is vertical and has unit length it extends from $(x, y, 0)$ to $(x, y, 1)$. Because all triangle in S have z -coordinate between 0 and 0.5 none of them can contain the point (x, y) . Hence, the point (x, y) in t is not covered.

9. Conclusions

In this paper we have identified a large number of geometric problems for which it is impossible to obtain subquadratic algorithms, unless one manages to improve the

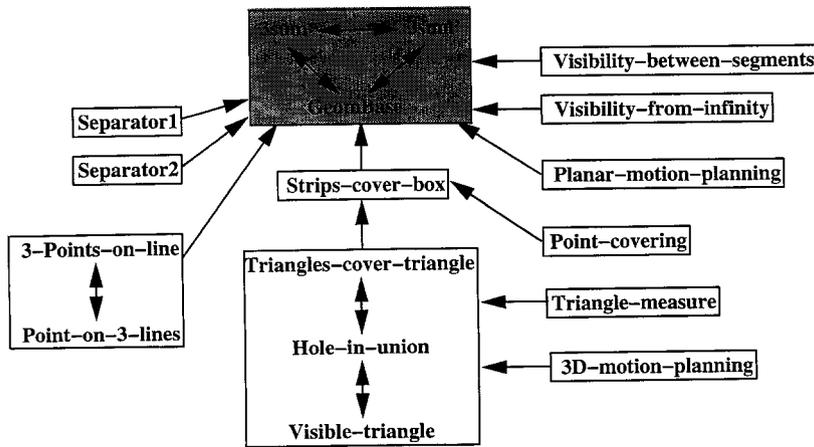


Fig. 11. Overview of the different relations.

complexity of the very simple formulated base problem 3SUM. It should be stressed though that all reductions presented in this paper do not prove anything on the time complexity of the different problems. They only show a relation between complexities but, because no lower bounds are known for 3SUM, these reductions have no immediate implications. An overview of the relations between the different problems is given in Fig. 11 where an arrow from problem A to problem B means that B can be solved using A (that is, B is ‘easier’ than A).

The notion of 3SUM-hard problems has helped us quite a bit in the recent past. Very often we have stumbled into 3SUM-hard problems and knowledge about them saved us the work of trying to improve the time bounds to subquadratic. We expect that others will benefit from this as well.

An important question of course is what to do when it turns out you have run into an 3SUM-hard problem. There are a number of directions to go. One way is to try and find output sensitive solutions. For example, as noted before, one can extend the problem 3-POINTS-ON-LINE to K-POINTS-ON-LINE: Given a set of points in the plane, is there a line containing at least k of the points. In [13] it is shown that this problem can be solved in time $O((n^2/k) \log n/k)$ which is much faster than $O(n^2)$ if k grows with n . Another solution might be to look at algorithms that work efficiently for special classes of objects. In particular so-called ‘fat’ objects seem to provide big improvements. (Objects are called fat if they don’t have long thin parts. See [20] for a precise definition). For example, the problem HOLE-IN-UNION can be solved in time $O(n \log n)$ when the triangles are fat (see [16]). Also the problem PLANAR-MOTION-PLANNING can be solved in time $O(n \log n)$ for a rod moving among fat obstacles (see [21]). Efficient solutions might also exist for other types of special cases. For example, in [10] it is shown that problem SEPARATOR2 can be solved in time $O(n \log n)$ when the size of the smallest segment is at least some constant fraction of the diameter of the set of segments.

The major open problem that remains is to get better bounds on the complexity of the problem 3SUM. Almost any improvement in the lower bound would immediately apply to all other problems because the reductions normally take very little time. On the other hand, an improvement of the upper bound would be essential to improve the time bounds of the other problems studied. As indicated before, recently Erickson and Seidel [11] have obtained an $\Omega(n^2)$ lower bound for the 3SUM problem but only under a weak model of computation. It should though be said that almost all “natural” algorithms for the problems in question fit within this model.

The other direction of research is of course to extend the class of 3SUM-hard problems. In this paper we have tried to concentrate on a number of rather general problems. We have tried to formulate the simplest versions of the problems that are already 3SUM-hard. As indicated at many places this immediately proves that more complicated versions of the same problem are 3SUM-hard as well. We are though convinced that many more problems fall in this class. For example, in a recent paper ([2]) it is shown that particular versions of injection molding are 3SUM-hard. Also, in [4] it is shown that the computation of perfect binary space partition trees is 3SUM-hard.

Finally, other interesting classes of related problems might exist. A first direction would be to consider problems with n^d bounds for $d > 2$. It seems though difficult to generalize the approach described in this paper. Also, there is a large class of problems for which $\Theta(n^{4/3})$ seems to be the best achievable time bound. Study of such classes might provide insight in the intrinsic complexity of certain types of problems.

Acknowledgments

The results reported in this paper are definitely not the sole contribution of the authors. Many researchers have contributed to this work by proposing 3SUM-hard problems, by suggesting reductions, etc. We would like to acknowledge the contributions by Helmut Alt, Tetsuo Asano, Bernard Chazelle, Mark de Berg, Herbert Edelsbrunner, Leo Guibas, Dan Halperin, Jiří Matoušek, Marco Pellegrini, Otfried Schwarzkopf, Micha Sharir, Jack Snoeyink, Paul Spirakis, Marc van Kreveld, and anybody else we might have forgotten.

References

- [1] S. Bloch, J. Buss and J. Goldsmith, How hard are n^2 -hard problems?, SIGACT news 25 (1994) 83–85.
- [2] P. Bose, M. van Kreveld and G. Toussaint, Filling polyhedral molds, Proc. 3rd Workshop Algorithms Data Struct., Lecture Notes in Computer Science 709, (Springer, New York, 1993) 210–221.
- [3] M. de Berg, Generalized hidden surface removal, Proc. 9th ACM Symp. on Computational Geometry (1993) 1–10.
- [4] M. de Berg, M. de Groot and M. Overmars, Perfect binary space partition trees, Proc. 5th Canad. Conf. Comput. Geom. (1993) 109–114.

- [5] B. Chazelle, Triangulating a simple polygon in linear time, *Discrete Comput. Geom.* 6 (1991), 485–524.
- [6] H. Edelsbrunner, *Algorithms in Combinatorial Geometry* (Springer, New York, 1987).
- [7] H. Edelsbrunner and L. Guibas, Topologically sweeping an arrangement, *J. Comp. Syst. Sc.* 38 (1989) 165–194.
- [8] H. Edelsbrunner, L. Guibas and M. Sharir, The complexity and construction of many faces in arrangements of lines and of segments, *Discrete Comp. Geom.* 5 (1990), 161–196.
- [9] H. Edelsbrunner, M. Overmars and D. Wood, Graphics in Flatland: a case study, in: F.P. Preparata, ed., *Advances in Computing Research, Vol 1: Computational Geometry* (JAI Press, 1983) 35–59.
- [10] A. Efrat, M. Sharir and G. Rote, On the union of fat wedges and separating a collection of segments by a line, *Comput. Geom. Theory Appl.* 3 (1994) 277–288.
- [11] J. Erickson and R. Seidel, Better lower bounds on detecting affine and spherical degeneracies, *Proc. 34th Annu. IEEE Sympos. Found. Comput. Sci. (FOCS 93)* (1993) 528–536.
- [12] R. Freimer, J. Mitchell and C. Piatko, On the complexity of shattering using arrangements, *Proc. 2nd Canad. Conf. Comput. Geom.* (1990) 218–222.
- [13] L. Guibas, M. Overmars and J.-M. Robert, The exact fitting problem for points, *Proc. Third Canad. Conf. Comput. Geom.* (1991) 171–174.
- [14] J.-C. Latombe, *Robot Motion Planning* (Kluwer Academic Publishers, Norwell, 1991).
- [15] Y. Ke and J. O'Rourke, An algorithm for moving a ladder in three dimensions, Technical Report JHU-87/17, Department of Computer Science, Johns Hopkins University, 1987.
- [16] J. Matoušek, J. Pach, M. Sharir, S. Sifrony and E. Welzl, Fat triangles determine linearly many holes, *SIAM J. Comput.* 23 (1994).
- [17] M. McKenna, Worst-case optimal hidden surface removal, *ACM Trans. Graphics* 6 (1987) 19–28.
- [18] J. O'Rourke, Computational geometry column 22, *SIGACT news* 25 (1994) 31–33.
- [19] F. Preparata and M. Shamos, *Computational Geometry, An Introduction*, (Springer, New York, 1985).
- [20] A. van der Stappen, D. Halperin and M. Overmars, The complexity of the free space for a robot moving amidst fat obstacles, *Comput. Geom. Theory Appl.* 3 (1993) 353–373.
- [21] A. van der Stappen, D. Halperin and M. Overmars, Efficient algorithms for exact motion planning amidst fat obstacles, *Proc. IEEE Intern. Conf. on Robotics and Automation, Vol. I* (1993) 297–304.
- [22] G. Vegter, The visibility diagram: a data structure for visibility and motion planning problems in the plane, *Proc. SWAT'90, Lect. Notes in Comp. Science* 447 (Springer, Berlin, 1990) 97–110.
- [23] E. Welzl, Constructing the visibility graph for n line segments in $O(n^2)$ time, *Inform. Process. Lett.* 20 (1985) 167–171.